Journal of Universal Computer Science, vol. 13, no. 2 (2007), 267-286
submitted: 31/7/06, accepted: 15/1/07, appeared: 28/2/07 © J.UCS

Using Recommendation to Improve Negotiations in Agent-based Systems

Mateusz Lenar
(Institute of Applied Informatics, Wroclaw University of Technology Poland
lenar@pwr.wroc.pl)

Janusz Sobecki
(Institute of Applied Informatics, Wroclaw University of Technology Poland
sobecki@pwr.wroc.pl)

Abstract: In this paper we present research works on non-intuitive and low-efficient negotiations between agents in agent based system. We find recommendation techniques as a suitable method of making negotiation smarter and more efficient. We have introduced into the negotiation thread the improvements in the form of hybrid recommendation. Hybrid recommendation is composed of three basic elements: demographic, collaborative and content-based. On the base of a negotiation algorithm we have studied how recommendation methods can improve the whole process of finding mutually acceptable agreements between agents. The proposed methodology is presented using a travel agency and its client negotiation.

Keywords: negotiation thread, Petri Nets, hybrid recommendation, agent-based systems
Categories: D.2.2, H.5.2, H.5.3, I.2.11

1 Introduction

A still growing computational power of computers causes users to expect more from applications. “More” means not only faster and unfailingly but also intelligently and adaptively. In recent years we can observe a rapid development in the field of applied artificial intelligence. AI techniques allows developers to create software to be more “human-like”. Our idea is to employ recommendation mechanisms into negotiation in agent-based systems. What is an agent? Why we use agent approach? In our approach agent is treated, like in a real world, as a representative of some user’s interests. Agent-based environments are the collection of many agents created similarly to human community. Agent based system can be defined as a common platform where autonomous agents are running concurrently and independently to achieve their individual goals. Like human beings in real world, agents have to communicate with each other in the understandable way. Moreover, the communication language should be as simple as possible to make conversation easy and short. But fast and understandable conversation is not sufficient to achieve agents’ goals. In agent based system the agents’ goals are often contradictory and the designer’s role is to build such a mechanism, which not only will resolve the conflicts but also do it in efficient and smart way. This is a purpose why we introduced the negotiation techniques modified by recommendation methods into communication process.

In [Section 2] the negotiation as a method in agent-based systems' conflict resolving is presented. In [Section 3] the recommender systems and recommendation methods are described. The following [Section 4] describes the travel-agent negotiation system ontology and its architecture. In [Section 5] the negotiation algorithm with application of recommendation methods is presented. In [Section 6] the case study for travel agency is shown. The last [Section 7] concludes the paper and shows perspectives for the future works.

2 Negotiation as a Method in Agent-based Systems' Conflict Resolving

Etymology of a term "negotiation" is strictly connected with trading. In Latin "*negotiari*" means to take care about a business. From ancient times people use negotiation techniques in different fields of human life: to bargain on the market, to resolve real-world conflicts, to establish rules of cooperation and even unconsciously in such everyday situations like walking on park's paths or pavements with other people. According to the definition formulated by Nicolas Jennings, negotiation in agent based systems is a process, "by which group of agents communicate with one another to try and come to a mutually acceptable agreement on some matter" [Jennings et al. 00]. We employed negotiation as a form of extended communication in agent based system because these systems are trying to work as human beings and like in real world, negotiation seems to be a natural and intuitive way of communication.

2.1 Basic notions in the consensus model

"Conflict" may be defined as interests' contradiction or as an effect of different perception of the same subject. In computer science conflicts are common e.g. in distributed databases or in computer networks. Even in not very large agent-based systems conflicts are also unavoidable because of autonomy of interacting agents, distribution of interacting agents; lack of centralized control and limited resources or even lack of resources. Each autonomous agent determines its own goal, which is often in conflict with goals of other agents [Barber et al. 99]. Moreover, the knowledge of the same subject is widely distributed and varies from each other. This knowledge may also be stored in various ways and often has different representation. Agents have to interact very frequently because of incomplete and uncertain environment and compete with each other for limited resources. In general, conflicts may be resolved e.g. using consensus methods [Nguyen 01], [Nguyen 02] or through negotiations [Lenar and Zgrzywa 04].

There are three basic levels of conflict classification. These levels determine different types of conflict situation. These conflicts are quite similar to real world ones e.g. marathon, road traffic. Goal conflict appears when agents' goals can not be reached simultaneously. An agent or a supervisor should redefine the goal by making conditions weaker. Resource conflict exists when agents' plans of using common resources are contradictory. Then the sequence of resource reservations should be rearranged to avoid system deadlock. Finally, the last level of conflicts comes with agents' knowledge incompleteness or skills insufficiency. An agent does not have enough inner knowledge or skills to independently and completely solve the tasks and it has to cooperate with other agents.

2.2 Negotiation Mechanism

Negotiation mechanism consists of protocols, strategies and possible deals. Negotiation mechanism should be designed and judged by the following criteria [Zlotkin and Rosenschein 93]:

- adequacy - has to be provided both in quantitative and qualitative sense. Negotiation mechanism has to be constructed in complete but simple way, giving all agents possibility to act in several ways. Moreover, every agent should be treated equally.
- efficiency - negotiation mechanism has to be Pareto optimal,
- self-motivation – some motivating features should be provided to increase agents' will to co-operate,
- simplicity – communication language and negotiation mechanism have to be designed in simple way, should be easy to implement and to understand.

2.3 Negotiation Protocol

Negotiation protocol is an essential part of negotiation mechanism. The protocol is described by the principles and rules of an interaction between agents. The protocol is designed and implemented on the base of existing lower-level communication protocols. It is focused on definition of communication semantics. In the protocol we specify:

- number and types of participants;
- reachable system states;
- rules specifying how to change the state;
- sequence of making offers;
- types of possible negotiation deals.

2.4 Negotiation Strategy

The negotiation strategy is a set of rules and functions that are used in the negotiation process. The strategy covers aims and manners to reach the mutually acceptable state of the system. There are three basic types of negotiation strategy: resource-based strategy, time-dependent strategy and imitating strategy. We focus on a resource-dependent strategy, where an agent weakens its negotiation position not below specified level in a specified pace. Formula 1 describes a typical contract scoring function used in resource dependent strategies. c_1, c_2, c_3 and c_4 are constants.

$$F(t_i) = c_1 e^{c_2(t_i - c_3)} + c_4 \tag{1}$$

2.5 Formal Model of Negotiation

The formal model of negotiation contains: sets of agents and issues, and a negotiation thread [Faratin et al. 97].
$A = \{a_1, a_2, .., a_n\}$ – a finite set of negotiating agents,
$I = \{i_1, i_2..., i_m\}$ – a finite set of issues being under negotiation, e.g. price, colour, delivery time. The issues are numbered from 1 to m. Each issue may receive continuous values (e.g. delivery time) or discreet values (e.g. colour names).
$a \in A$ – negotiating agent,
$i \in I$ – issue being under negotiation,
$x_i \in [min_i, max_i]$ – real value of continuous issue i,
$x_i \in \{value_{i1}, value_{i2}, ..., value_{ik}\}$ – a set of real values of discreet issue i,
min_i, – minimal value of issue i,
max_i – maximum value of issue i,
$F_i^a : [min_i, max_i] \rightarrow [0, 1]$ – scoring function, that evaluates the score of continuous issue value.
$F_i^a : \{value_{i1}, value_{i2}, ..., value_{ik}\} \rightarrow [0, 1]$ – scoring function, that evaluates the score of discreet issue value.
w_i^a - relative importance of issue *i* to agent *a* (the issues are weighted in case of their importance)
$\sum_i w_i^a = 1$ - weights for each set of issues are normalized.
$F^a = \sum_i w_i^a \cdot F_i^a(x_i)$ - agent scoring function for a contract.
$x_{a \rightarrow b}^t$ - one dimensional indexed offer vector generated by an agent *a* at the moment of time *t* and sent to an agent *b*.

The scoring function value at the specific moment of time should be as close as possible to the value of expected score of the contract F^a that is calculated using contract scoring function $F^a(t_i)$. In most cases there is necessity to meet the following criterion:

$$F^a \leq F^a(t_i) \tag{2}$$

The bilateral negotiation thread between agents a and b is any finite sequence of the form $\{ x_{a \rightarrow b}^{t_1}, x_{b \rightarrow a}^{t_2}, ..., x_{a \rightarrow b}^{t_n} \}$, where $t_1 < t_2 < ... < t_n$. Interpretation of an offer vector x by an agent a at the time *t' > t* using scoring function F^a is:

- accepted, when formula 3 is true;

$$F^a(x_{b \rightarrow a}^{t_k}) \geq F^a(x_{a \rightarrow b}^{t_{k-1}}) \tag{3}$$

- rejected, when there is reached compromise in the other concurrent thread, or one of the negotiation issues has reached its critical values (e. g. time deadline was reached);
- active, when formula 4 is true and there is generated a counteroffer $x^{t_{k+1}}_{a\to b}$.

$$F^a(x^{t_k}_{b\to a}) < F^a(x^{t_{k-1}}_{a\to b}) \tag{4}$$

In other words, the negotiation thread is active until agreement is set or until a deadline is reached. Every unsatisfying offer forces agent to create a counteroffer that is still acceptable for the agent and that should be closer to other agent's requirements. The scoring function value in the next step should not be greater than in the previous step. In most cases the value in the next step is lower. According to the formula 4 the agent should create counteroffer using some algorithm. The details of using recommendation techniques will be described in subsequent sections. After the counteroffer is created we have to check if the criterion given in formula 4 is true or not. When it is true then the agent can send the offer, but when it is not then the agent has to modify the offer to meet the criterion or choose another offer from recommendation mechanism.

We use bilateral negotiation approach to make the negotiation simpler. Of course, at one time an agent can run many negotiation threads. The synchronization and coordination mechanisms are required. At the design and simulation level we use Petri nets to avoid deadlocks. Similar techniques are used at the implementation and runtime level.

3 Recommender Systems

With growing popularity of different web-based systems recommendation methods help to deliver customized information to a great variety of users. Recommender systems may be applied in many different domains, such as [Montaner et al. 03]: netnews filtering (ACR News, SWIT Netnews, News Dude), web recommender (Letizia, ifWeb, WebWatcher), personalized newspaper (Anatagonomy, Krakatoa Chronicle), sharing news (Beehive), movie recommender (MovieLens, Recommender), document recommender (Casmir), information recommender (InfoFinder), E-commerce (Amazon, CD-now), purchase, travel and store recommender (LifeStyle Finder), E-mail filtering (Re:Agent), music recommender (Ringo/FireFly) and music list recommender (Smart Radio).

The central problem of recommender systems is user modelling [Kobsa et al. 01]. The user model concerns usually data: content, representation and utilization within the systems. The data is divided into two main parts: user data that characterizes the user itself and usage data that concerns different aspects of the user interaction with the system. The user data contains information on demographic data, users' knowledge, their skills and capabilities, their interests and preferences and also their plans and goals. The usage data may concern selective operations that express users' interests, unfamiliarity or preferences, temporal viewing behaviour, as well as ratings concerning the relevance of these elements.

The model representation and utilization is further addressed by Montaner in [Montaner et al. 03], who introduces the taxonomy of recommender agents that distinguishes two dimensions: profile generation & maintenance and profile exploitation. The former contains the following elements: user profile generation, initial profile generation, profile learning technique and relevance feedback. The later contains: information filtering method, user profile-item matching technique, user profile matching technique and profile adaptation technique. One of the most distinguishing dimension of the recommender systems is information filtering method that according to Mantaner may be: demographic (DF), content-based (CBF), collaborative (CF) and hybrid (HA). Other authors [Kazienko and Kiewra 04] present also following types: case-based reasoning (CBR), rule-based filtering (RBF). In this section we present the following filtering methods: DF, CBF, CF and HA in more details, and application of CBR and RBF will be only mentioned at the end of this section.

3.1 Demographic Filtering

DF uses a stereotype reasoning [Montaner et al. 03] in recommendations and is based on the information stored in the user profile that contains different demographic features. The demographic data that is an element of the user data, contains the following elements: record data (name, address, e-mail, etc.), geographic data (zip-code, city, state, country), user's characteristics (sex, education, occupation), and some other customer qualifying data.

Stereotype reasoning is a classification problem that is aimed at generating initial predictions about the user [Kobsa et al. 01]. For example zip-codes of living places may be sufficient to draw quite detailed assumptions on the people's social status, interests and various purchasing behaviors [Quinn and Pawasarat 01]. Also in the user interface recommender system [Sobecki and Weihberg 04] the demographic recommendation is based on stereotype reasoning. Initially the stereotypes were determined by the group of experts by specifying the centroids of demographic attributes values which represented several initial classes of users. These values should be selected in such a way that none of them had all the extreme (maximal or minimal) values and the distance between consecutive centroids was similar. For each of these centroids the corresponding interface profile was assigned by the expert, which then was recommended to the user after registering to the system by delivering demographic information.

In "The Cooking Assistant" described in [Sobecki et al. 06] the demographic recommendation applied a fuzzy logic inference. The main advantage of application of fuzzy reasoning over the standard stereotype reasoning is giving experts the more convenient tool for modeling different relationships from the real world. The general fuzzy inference system applied in this recommender system follows the general model presented in [Paplonski 04], where the particular input characteristics is mapped to input membership functions, which is mapped to rules that are mapped into a set of output characteristics and consequently to output membership function, which finally is mapped to a single-valued output associated with the decision.

In "The Cooking Assistant" we selected the following demographic attributes describing each user for the input: age, gender, number of inhabitants in the place of living; user knowledge attributes: cooking experience and preferences: vegetarian or not. The values of the selected demographic attributes were used as the input to determine the membership function values of linguistic variables. The selected attributes were also used to construct fuzzy inference rules that for some assumptions expressed in so called linguistic values (e.g. medium age) assign conclusions, in this case concerning cooking recipes description also expressed in the linguistic values. In the system we implemented over 170 fuzzy rules.

For the defuzzification we applied very simple method called Mean of Maximum (MOM), which takes the mean value of the set with maximum membership grade. According to the fuzzy inference system we determine values that represent cooking recipes: difficulty, fantasy and number of calories. Then using standard cosine similarity function we determine the most similar recipes with the value over the specified limit. These recipes are sorted with descending similarity value and the number is limited to 15 recipes.

The demographic recommendations have however some disadvantages [Montaner et al. 03], [Nguyen and Sobecki 03]:

- for many users generalizations of the user's interests associated with some demographic attribute values may be too general;
- they do not provide any individual adaptation, also when the user interests tend to change over time;
- users are quite often reluctant to submit demographic information or lie in this matter.

3.2 Content-based Filtering

CBF takes descriptions of the content of the previously evaluated items to learn the relationship between a single user and the description of the new items [Montaner et al. 03]. We can find quite many applications of interface agents, for example, Letizia, an autonomous interface agent [Fleming and Cohen 99] for Web browsing [Lieberman 97]. Letizia constructs the user profile out of the recorded URL's of visited pages. Then, using simple keyword-frequency measure, adopted from the field of Information Retrieval, the agent searches the neighborhood of pages currently visited for potentially relevant pages. Another interface agent is Apt Decision that learns user's preferences in the area of the real estates rental in order to suggest appropriate apartments [Shearin and Lieberman 01]. Apt Decision agent uses initial profile provided by the user as well as descriptions of apartments extracted from offers the user has analyzed so far. In the same way CBF may be used in the area of tourist travel recommendation.

However CBF approach enables personalized and effective recommendations for particular users, but has also some disadvantages:

- content-based approaches depend on so called objective description of the recommended items;
- it tends to overspecialize its recommendations;
- content-based approach is based only on the particular user relevance evaluations, but users usually are very reluctant to give them explicit, so usually other implicit, possibly less adequate, methods must be used.

3.3 Collaborative Filtering

CF makes automatic predictions (filtering) about the recommended items by collecting and using information about testes of other users (collaboration) [wikipedia 06]. CF method is very popular among many movie and music recommender systems. Most of the CF based recommender systems are using user item rating matrix that is used for both: identifying similar users and recommend items highly ranked by those users. The other approach called also item-based approach uses item-item matrix to determine the current user taste according to selection one item. This approach is used for example in Amazon.

In the recommender system presented in [Sobecki and Weihberg 04] the user interface is recommended according to the values of demographic attributes as in DF, however the specified user interface settings values are determined using the consensus method [Nguyen and Sobecki 03]. The consensus is determined among all the user interface settings values that were given by the group of similar users.

The main advantages of collaborative filtering over the content-based architecture are following [Montaner et al. 03]:

- the community of users can deliver subjective data about items;
- collaborative filtering is able to offer novel items, even such that user have never seen before;
- collaborative recommendation utilizes item ratings of other users to find the best fitting one.

Collaborative recommended agents have also some disadvantages:

- when the number of other similar users is small then the prediction is rather poor;
- the quality of service for users of peculiar tests is also bad; this is rather difficult to get sufficient number of similar users to be able to make proper predictions; observe their users and then apply some machine learning mechanisms to draw the recommendation
- lack of transparency in the process of prediction and finally the user's personal dislike may be overcome by the number of other similar users opinions.

The drawbacks of CF could overcome by applying the hybrid solution, for user interface recommendation for web-based information system presented in [Sobecki and Weihberg 04] the disadvantages mentioned above do not influence it much. First, we can assume that web-based systems always have quite many similar users. Second, when the prediction does not fit the user, he is able to personalize the interface manually.

3.4 Hybrid Approach

The disadvantages of each of the above mentioned recommendation approaches could be overcome by applying HA. For example the disadvantage of the insufficient number of the similar users at the early stages of the system operation using CF may be overcome by application of the demographic stereotype reasoning.

For example in the user interface recommendation was based on the mixture of the DF and CF [Nguyen and Sobecki 03], [Sobecki and Weihberg 04]. Basically the HA [Sobecki 04] is a combination of demographic, collaborative and content based recommendation. However other types of recommendations that are based on: user emotions, user platform or context of use may be also considered.

We can distinguish at least two types of the HA, first that builds the recommendation basing on each single approach and second, that integrates the knowledge from each single approach before determining the recommendation. For example in the movie recommendation using the former approach we should first determine the lists of recommended movies using DF, CF and CBF separately and then combine these three lists. Using the later approach we modify the stereotype reasoning or fuzzy reasoning rules according to the data or rules of other users (CF) or ranked movies (CBF). Then the final recommendation is determined using these modified rules.

3.5 Other Recommendation Approaches

Beside above mentioned recommendations: DF, CF, CBF and HA, we should also mention other ones, such as: platform, situation or emotion based, which could be combined under single name case-based reasoning (CBR). These recommendations may be dealt in two different ways. The first one is based on the expansion of the subject's attribute set with the attribute concerning platform, situation or emotions in standard DF, CF or CBF. The second method treats these recommendations as separate ones, with their own knowledge acquisition methods and reasoning rules.

4 System Ontology and Architecture

Ontology consists of terms, definitions and axioms. So to employ negotiation and recommendation into the process of holiday reservation we have to define domain ontology: The basics of the ontology are containing definitions of the following elements: negotiated objects' models (i.e. holiday travels); negotiation participants (travel agent, customer); distances between negotiated objects and customers; rules of interaction.

4.1 Negotiated Object's Model

Definition of negotiated object consists of the specification of issues and their values of the single distinguished object, i.e. a holiday travel. Each holiday travel is described in form by a tuple of values $t : A^h \to V^h$, where A^h denotes set of attributes describing holiday tour and V^h denotes set of sets of values of these attributes. Attributes and their values are described below in Figure 1. The values are enumerated or the type of values is given.

Holiday travel

Customer preferences

Offer-id	Unique identifier
Destination	Set={Egypt-Hurgada, Egypt-Sharm el Sheik, Spain-Gran Canaria, Spain-Tererife, Cyprus-Larnaca, Tunisia-Tunis}
Departure-Airport	Set={Warsaw, Cracow, Wroclaw, Poznan, Katovice, Gdansk, Szczecin}
Hotel-ratings	Set={5 or higher, 4 or higher, 3 or higher, 2 or higher, 1 or higher}
Food	Set={HB, FB, None}
Hotel-amenities	Vector=(fitness room, swimming pool, kid's activities, game room, golf course, dining room, tennis courts, pets allowed, internet access, diving, windsurfing, hotel beach)
Room-facilities	Vector=(bathroom, WC, mini-bar, TV, air-condition, balcony or terrace)
Room-type	Set={single, twin, double, triple, four person}
Number-of-adult-travelmates	Integer number
Number-of-children-under-15-years-old	Integer number
Start-of-the-holiday	date
Duration	Integer number
Price	Real number
Early-booking-discount	Percent
Commission	Real number

Figure 1. Conceptualization of holiday travel and customer preferences

4.2 Negotiation Participants

Within the negotiation ontology we can distinguish two different participants: a travel agent and a customer one. The customer agent's description consists of three elements: a tuple of customer's demographic data, a tuple of customer's preferences and a customer's usage data in form of a list of former holidays together with their evaluation (from 0 to 5). A tuple of customer's demographic data is represented as $t: A^c \rightarrow V^c$, where A^c denotes set of attributes customer's demographic data and V^c denotes set of sets of values of these attributes. Attributes and their values are described below in Figure 2. A tuple of customers preferences is represented as a holiday travel tuple (as described above). Former holidays are also described in the same way.

The travel agent's description contains: travel agent preferences including minimal commission and an agent usage data in the form of a history of the former negotiations.

Customer Demographic data

Age	Integer number
Gender	Set={Female, Male}
Address-zip-code	Zip code
Number-of-adult-travelmates	Integer number

Figure 2. Conceptualization of customer demographic data

4.3 Distance Definition Between Holiday's Descriptions and Customer's Profiles

Both holiday's description and customer's profiles are described by corresponding tuples of attributes values. The distance function between values of these attributes is defined as a function $\delta^a : V_a \times V_a \rightarrow [0,1]$ where V_a denotes V_a^h or V_a^c (set of holiday's or customer's values of an attribute) for all $a \in A^h$ or $a \in A^c$. These functions are given by the system designer to distinguish specific properties of each attribute. These functions should fulfill all the distance function conditions but not especially all the metrics conditions. The distance function values could be enumerated (for example distance between different holiday destinations) or given in any procedural form (for example distance between holiday prices).

The distance between descriptions (holiday's or customer's) could be defined in many different ways, however we propose to define the distance between tuples *i* and *j* as a weighted sum of distances between values of each attribute. The weight indicates the importance of each attribute *a* by multiplying the distance by appropriate factor defined as a function $c : A \rightarrow [0,1]$ and $\sum_{a \in A} c(a) = 1$, so finally the distance between tuples is defined as follows:

$$\delta(r_i, r_j) = \sum_{a \in A} [c(a) * \delta^{at}(r_i(a), r_j(a))] \tag{5}$$

4.4 Customer Clustering Based on the Customer Profile

Clustering problem is defined as a partition of the given set of customer profiles into subsets such that a specific criterion is optimized. The criterion is often defined as the distance between a profile and the corresponding cluster center. To minimize this criterion we use k-means clustering that partitions the set of the profiles into k non-overlapping clusters that are identified by their centers.

However k-means problem is known to be NP-hard. There are quite many sub-optimal solutions to that problem that have polynomial computational complexity. One of the simple and flexible solutions to this problem, i.e. Lloyd's algorithm [Kanungo et al. 02]. The steps of this algorithm are as follows. First, select randomly *k* elements as the starting centers of the clusters (centroides). Second, assign each element of the set to a cluster according to the smallest distance to its centroid. Third, recompute the centroid of each cluster, for example the average of the cluster's elements. Fourth, repeat steps 2 and 3 until some convergence conditions have not been met (for example centroides do not change).

The attractiveness of Lloyd's algorithm lies not only in its simplicity but also its ability to terminate when using the above mentioned convergence condition and for configurations without equidistant elements to more than one centroid. However its computational complexity is polynomial, the Lloyd's algorithm is quite time consuming. First, the step 2 that has to be performed in each iteration costs $O(kdN)$, where *d* is the dimension of each element and *N* is the number of elements. Second, algorithm usually needs many iterations to terminate. In the literature we can find quite many modification of this algorithm that run faster, for example bisecting k-means.

4.5 Application of Consensus Methods in the Holiday Travel Negotiation

The consensus methods are used to collaborative recommendation of holiday travels for the particular customer. In order to do this we must have the group of similar customers. The grouping may be done by means of k-means algorithm according to the customers profile and/or holiday travels they had before and ranked them pretty high. The customers group will be denoted as *G*.

The holiday travels that were highly ranked are also used for the consensus determination. Let *j* be the index of the group *G* member, r_j be the tuple that describes the holiday travel that was highly ranked by the customer *j*. Then to find the consensus (recommended for the particular group of customers *G*) we must find the holiday travel description *r* that conform the following formula [Nguyen 01]:

$$min(\sum_{j\in G}\delta(r,r_j)) \tag{6}$$

This problem could be computationally difficult, but according to [Nguyen 01] we can reduce the computation by finding the minimal value for all attributes a of a tuple separately:

$$min(\sum_{j\in G}\delta(r(a),r_j(a))) \tag{7}$$

5 Negotiation Algorithm with Recommendation

The negotiation algorithm with recommendation is based on the following assumptions: the travel agent's goal is to maximize the commission and the customer agent's goal is to maximize the discount and to find travel offer closest to customer profile. Customer profile includes weights of each issue. The unwanted values have attached the lowest value. Not evaluated issues have no weights; however they may be determined by means of the content-based recommendation. The customer may mark some issues that cannot be changed during negotiation. The customer sets also the weights of issues (destination – 4, hotel – 2 etc) and the weights of issue elements (*Spain-Gran Canaria* – 3, *Egypt-Hurgada* – 1 etc). Issues and issues' weights are normalized. Based on weights and values of issues there is calculated the "best offer", which in the algorithm plays a role of a model offer. The customer sets conditions for the "worst satisfying" offer, which is calculated in percent of the "best offer" value. This value gives the customer's agent to know about negotiation threshold. The simplified negotiation algorithm, which is applied both to customer and travel agent is presented below:

- *Step 1.* The agent creates the "best offer" and sends it to the other agent. Agent waits for an answer or for the negotiation deadline. Go to step 2.
- *Step 2.* If the negotiation deadline is not reached go to step 3 otherwise go to step 7
- *Step 3.* Evaluate received offer using agent's own scoring function. If the offer is acceptable go to step 4 otherwise go to step 5
- *Step 4.* The agent sends "accept" message. End of negotiation with success.
- *Step 5.* Create counter offer, sent it to the other agent and wait for an answer or for the negotiation deadline. Go to step 2

- *Step 6.* Send “reject” message. End of negotiation with failure.

We employ recommendation in steps 3 and 5. To evaluate the offer (step 3), customer agent uses its own scoring function (defined in paragraph 1) and content based recommendation. This will make this evaluation more realistic, because customer preferences are often incomplete.

To create the counteroffer (step 5) the customer agent should consider:

- lower demands
- creating a closer offer to the counteroffer using a distance function value
- an offer which is closer to the previous customer well evaluated travels
- a lower discount

To create the counteroffer (step 5) the travel agent should consider:

- a closer offer to the counteroffer based on customer clustering
- reducing a distance function value between created offer and the counteroffer
- preferring an offer which is closer to the previous customer well evaluated travels
- a lower commission

The negotiation strategy establishes the pace of reducing of the scoring function value. The algorithm is presented in the form of block diagram in Figure 3 where agent A is representing a travel agent and agent B is representing a customer agent.

Unfortunately, block diagram semantics has weak expression power to show the parallelism and synchronization of actions taken by independent agents. In that case we can see the same algorithm modelled using Petri nets, which are more suitable means to express the nature of interaction. Petri nets are concerned as a suitable method to simulate and model complex systems’ behaviours. They allow to model true concurrency and to describe system states and actions using its well-defined semantics. Next advantage of Petri nets is the ability to model multiagent system on different stages of system development. It starts from the simplest model where an agent is associated with the transition that occurred in the net. In this model whole Petri net is representing an entire multiagent system. During development process of the system more accurate and thorough specification should be necessary. In this situation one can create system model as a combination of autonomous nets. Each net can be modelled using specific Petri net, which describes behaviour of the net. It is very useful for developers for instance, because they receive a model only. They implement separately each class of agents using appropriate tools (there is no need to implement all classes of agents using the same software). The only restriction is that final agent software has to use the same ACL.

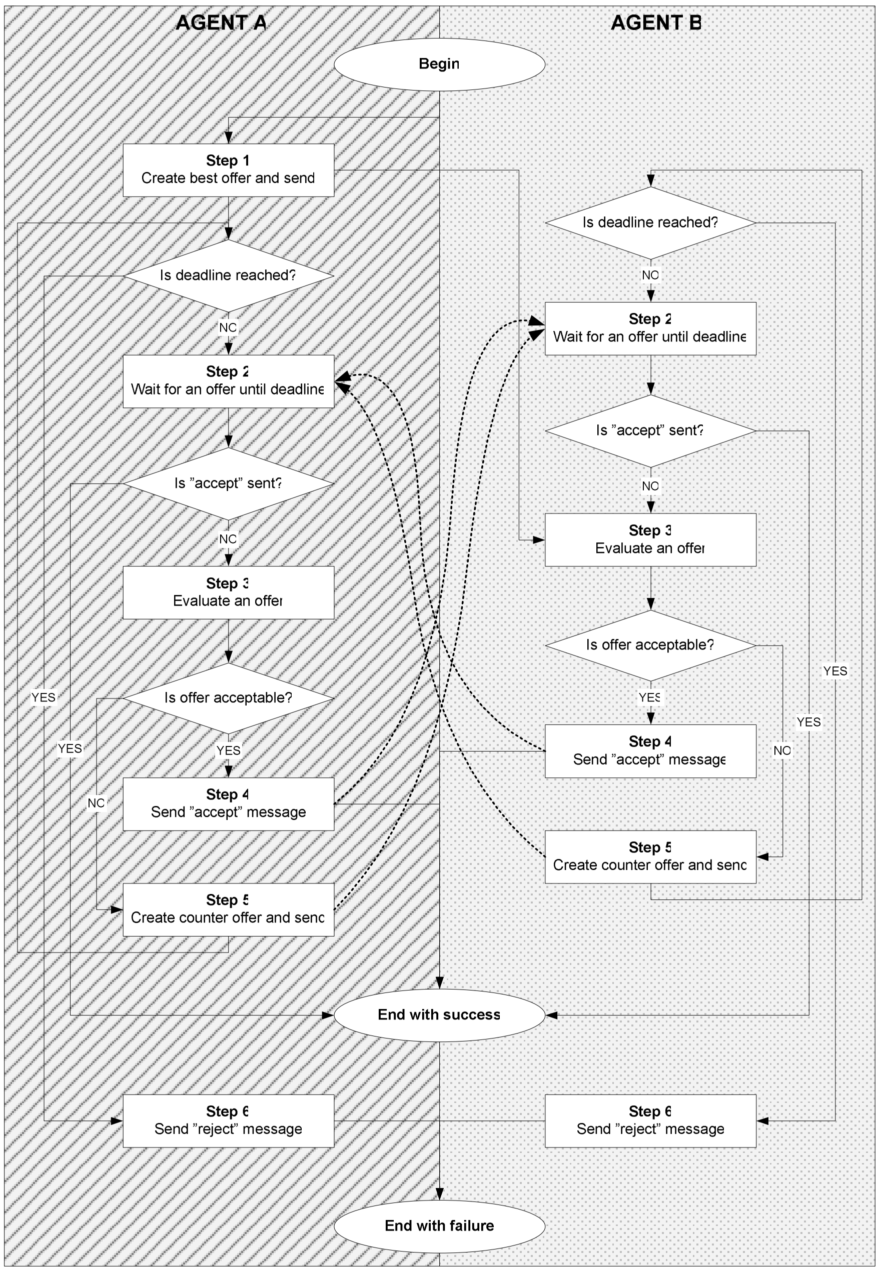

Figure 3. Negotiation algorithm block diagram

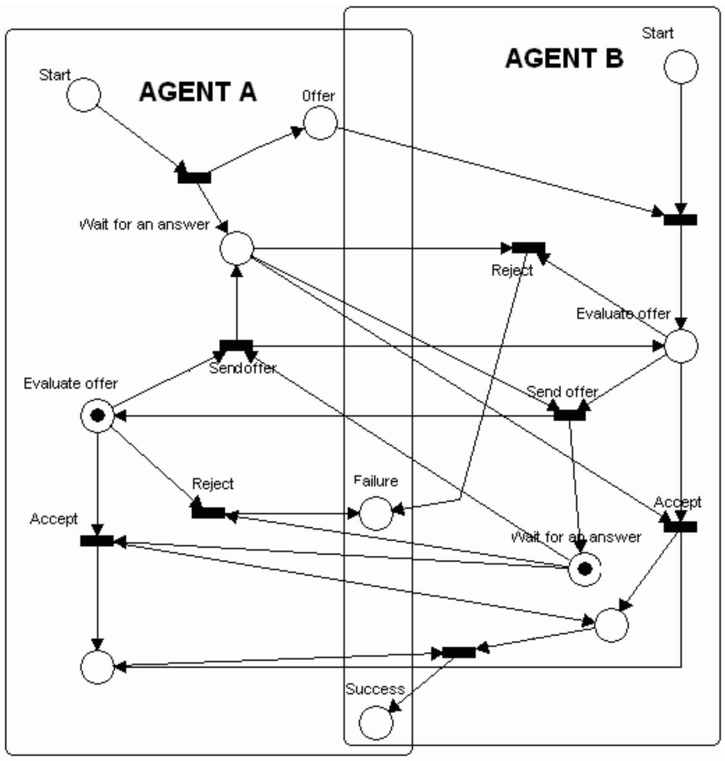

Figure 4. Negotiation algorithm described by means of Petri nets.

The most powerful advantages of high level Petri nets (HLPN) are:

- It can be used as a specification of the system or a presentation (because of a graphical representation),
- It has the ability to describe simultaneously states and actions using very few primitives,
- It integrates the description of data manipulation with the description of synchronisation and control,
- Its semantics allow to represent true concurrency,
- One can use abstraction and refinement (object, hierarchical),
- It support encapsulation and multiple use of shared code,
- There exist numerous tools supporting graphical modelling, simulating and analysing.

During last 10 years several frameworks and methodologies using HLPN have been developed to model and simulate multiagent systems [Fernandes and Belo 98], [Ferraro and Rogers 97], [Hiraishi 02], [Purvis et al. 02].

6 Case study

A customer agent has the profile (only non-zero weighted attributes are listed) presented in Table 1.

Issue *i*	w_i	F_i^a
Destination	0.10	Egypt-Hurgada→1.0 Egypt-Sharm el Sheik→1.0 Spain-Gran Canaria→0.8 Spain-Tererife→0.8 Cyprus-Larnaca→0.5 Tunisia-Tunis→0.2 Other→0.0
Food	0.05	HB→0.7 FB→1.0 None→0.3
Hotel-amenities fitness room, diving, windsurfing, hotel beach	 0.05 0.25 0.05 0.05	is supported→1.0 is not supported→0.0
Room-type	0.05	twin→1.0, double→1.0, other→0.0,
Number-of-participants	0.05	2→1.0 other→0.0
Start-of-the-holiday	0.10	[2006-06-01, 2006-06-07]→1.0 [2006-06-08, 2006-06-14]→0.8 [2006-06-15, 2006-06-21]→0.4 other→0.0
Duration	0.05	[7, 14]→1.0 [14, 21]→0.5 other→0.0
Price	0.20	[0, 1000]→1.0 [1000, 2000]→0.8 [2000, 3000]→0.5 [3000, +∞]→0.0

Table 1. Customer agent profile.

A recommendation mechanism creates the following counteroffer for the customer agent to evaluate:
Egypt-Sharm el Sheik, departure from Warsaw, Hotel 3 star, HB, swimming pool, dining room, internet access, diving, windsurfing, pets allowed, hotel beach,

bathroom, TV, air condition, double room, 2 persons, departure at 2006-06-03, arrival at 2006-06-10, total price 2700, early booking discount 100.

The scoring function value of the customer agent is (only non-zero weighted attributes are taken into consideration):

$$F^a = 0.10 \cdot 1.0 + 0.05 \cdot 0.7 + 0.05 \cdot 0.0 + 0.25 \cdot 1.0 + 0.05 \cdot 1.0 + 0.05 \cdot 1.0 + 0.05 \cdot 1.0 +$$

$$+ 0.05 \cdot 1.0 + 0.10 \cdot 1.0 + 0.05 \cdot 1.0 + 0.2 \cdot 0.5 = 0.835$$

We have designed several fuzzy functions e.g. "how far" (see figure 5). Argument of this function is a natural number which represents the distance in kilometres from the user's place of residence to the airport and the fuzzy values are "near", "on average distance" and "far".

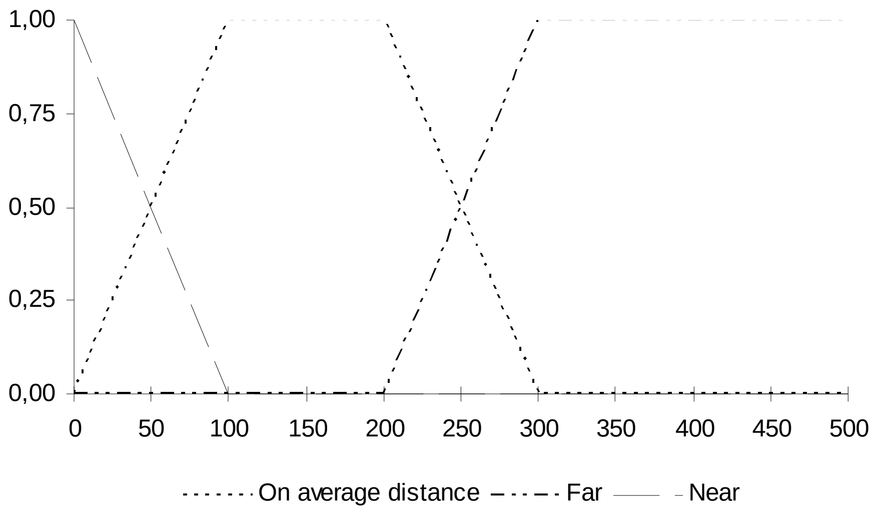

Figure 5. Fuzzy function "How far".

To transform some user or travel agent attributes into fuzzy function values we also need some auxiliary functions, such as $f_{zip} : ZIP \times ZIP \mapsto N$ which transforms Cartesian product of user's place of residence (represented by zip code) and airport zip code to a natural number telling how many kilometres is from user's place of residence to the airport.

All the fuzzy and auxiliary functions are implemented in relational database (using user defined functions and stored procedures). Also the fuzzy inference rules are also implemented in database using flexible meta descriptions that are easy to extend and maintain. Also user's and travel agent's descriptions are stored in the database.

During design process of inference rules we have made an assumption: the inference rule antecedent may contain several expressions joint by "AND" operator and the rule consequent may have only one expression.

We have also designed three sets of holiday travel attributes which describe three holiday profiles P: relax, sport and family (see Table 2). For every holiday travel we can find a fuzzy value for these three profiles (P∈ {Relax, Sport, Family}, HT means holiday travel).

$$F(P,HT) = \sum_{a \in A_P \wedge a \in HT} \frac{1}{card(A_P)} \quad \textbf{(8)}$$

Attribute	Relax	Sport	Family
fitness room	+	+	
swimming pool	+	+	+
kid's activities			+
game room	+		+
golf course	+	+	
dinning room			
tennis courts		+	
pets allowed			+
internet access	+	+	+
diving		+	
windsurfing		+	
hotel beach	+		+

Table 2. Holiday travel attributes mapping.

For example considering holiday travel to Egypt Hurghada, 3 star hotel with hotel beach, internet access, swimming pool, fitness, golf course, diving and windsurfing. Fuzzy values are following:

$$F(Relax, HT_{Egypt-Hurghada}) = 0.83$$

$$F(Sport, HT_{Egypt-Hurghada}) = 0.86$$

$$F(Family, HT_{Egypt-Hurghada}) = 0.50$$

7 Conclusions and Future Works

In this paper we presented extended version of the paper presented at the 9th International Conference on Knowledge-Based Intelligent Information and Engineering Systems KES 2005. However we have not managed to fully verify the proposed negotiation method with recommendation, we were able to show its preliminary results in the area of travel agency and its clients negotiation together with some examples. In the presented negotiation method we applied successfully Petri nets to model and simulate the negotiation thread. The recommendation method that was applied in the presented case study is mainly demographic character that takes into account the clients' holiday preferences, however it could be easily extended with dome demographic or collaborative components.

In our works that were published so far we have implemented and verify some elements such as negotiation thread control [Lenar and Zgrzywa 04] and fuzzy recommendation [Sobecki et al. 06]. Both elements showed a promising results, so we believe that its integration may show the effectiveness of the presented method.

The proposed negotiation methodology may be easily applied in other fields of e-commerce such as consumer electronics. However it is pretty easy to model negotiation for one shop and multiple clients, it would be also pretty interesting to create negotiation with recommendation for single client but multiple shops. In the near future we hope to finish both implementation of the presented method and its verification in the controlled environment. We hope that the application of the negotiation agents with recommendation will improve results of the negotiation process by making better and efficient deals for both sides.

References

[Barber et al. 99] Barber, K.S., Liu, T.H., Goel, A., Martin, C.E.: "Conflict Representation and Classification in a Domain-Independent Conflict Management Framework"; Proceedings of Third International Conference on Autonomous Agents, Seattle (1999), 346-347.

[Faratin et al. 97] Faratin, P., Jennings, N., Sierra, C.: "Negotiation Decision Functions for Autonomous Agents"; Elsevier Science (1997).

[Fernandes and Belo 98] Fernandes, J.M., Belo, O.: "Modeling Multi-Agent Systems Activities Through Colored Petri Nets: An Industrial Production System Case Study"; 16th IASTED International Conference on Applied Infomatics AI'98, Garmisch-Partenkirchen (1998), 17-20.

[Ferraro and Rogers 97] Ferraro, A, Rogers, E.H.: "Petri Nets in the Evaluation of Collaborative Systems"; IEEE First Int. Conf. on Systems, Man & Cybernetics, (1997).

[Fleming and Cohen 99] Fleming, M., Cohen, R.: "User Modelling in the Design of Interactive Interface Agents"; Proc of the 7th International Conference on User Modeling (1999), 67-76.

[Hiraishi 02] Hiraishi, K.: "PN^2 – An Elementary Model for Design and Analysis of Multi-Agent Systems"; 5th International Conference Coordination 2002, York (UK) (2002), 220-235.

[Jennings et al. 00] Jennings, N., Lomuscio, A., Wooldridge, M.: "Agent-Mediated Electronic Commerce", Springer (2000).

[Kanungo et al. 02] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C., Silverman, R., Wu, A.Y.: "An Efficient K-means Clustering Algorithm: Analysis and Implementation"; IEEE Tran. On Pattern Analysis And Machine Intelligence 24(7) (2002), 881-892.

[Kazienko and Kiewra 04] Kazienko, P., Kiewra, M.: "Personalized Recommendation of Web pages"; International Series on Advanced Intelligence 10 (2004), 163–183.

[Kobsa et al. 01] Kobsa, A., Koenemann, J., Pohl, W.: "Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships"; Knowledge Eng. Rev. 16(2) (2001), 111-155.

[Lenar and Zgrzywa 04] Lenar, M., Zgrzywa, A.: "Modelling Multi-Aspect Negotiations in Multiagent Systems Using Petri Nets"; Proceedings of 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2004, Ottawa, Canada, Lecture Notes in Computer Science 3029 Springer (2004), 199-208.

[Lieberman 97] Lieberman, H.: "Autonomous Interface Agents"; Proc. CHI 97, ACM (1997), 67-74.

[Montaner et al. 03] Montaner, M., Lopez, B., de la Rosa, J.P.: "A Taxonomy of Recommender Agents on the Internet"; Artificial Intelligence Review 19, (2003) 285-330.

[Nguyen 01] Nguyen, N. T.: “Conflict Profiles' Susceptibility to Consensus in Consensus Systems”; Bulletin of International Rough Sets Society 5(1/2) (2001), 217-224.

[Nguyen 02] Nguyen, N. T.: “Consensus Methods and Their Applications to Solving Conflicts in Distributed Systems”; Wroclaw University of Technology Press (2002), (in Polish).

[Nguyen and Sobecki 03] Nguyen, N. T., Sobecki, J.: “Using Consensus Methods to Construct Adaptive Interfaces in Multimodal Web-based Systems”; Universal Access in Inf. Society 2(4) (2003), 342-358.

[Paplonski 04] Paploński A. P.: “Neuro-Fuzzy Computing” (2006), Download date:10.03.2006 from http://www.csse.monash.edu.au/courseware/cse5301/04/Lnts/L12fz.pdf.

[Purvis et al. 02] Purvis, M., Nowostawski, M., Cranefield, S., Purvis, M., “Multi-Agent System Interaction Protocols in a Dynamically Changing Environment” (2002).

[Quinn and Pawasarat 01] Quinn, L.M., Pawasarat, J.: “Confronting Anti-Urban Marketing Stereotypes: A Milwaukee Economic Development Challenge June”, (2001), Download date: Dec 2002 from http://www.uwm.edu/Dept/ETI/purchasing/markets.htm.

[Shearin and Lieberman 01] Shearin, S., Lieberman, H.: “Intelligent Profiling by Example”; Proc of the Conf on Intelligent User Interfaces, ACM Press, (2001), Download date Dec. 2004 from http://web.media.mit.edu/%7Esibyl/projects/apt/final-shearin-iui.pdf.

[Sobecki and Weihberg 04] Sobecki, J., Weihberg, M.: “Consensus-based Adaptive User Interface Implementation in the Product Promotion”; In Keates S. et al.: "Design for a More Inclusive World", Springer-Verlag, London (2004), 111-121.

[Sobecki 04] Sobecki J.: “Hybrid Adaptation of Web-Based Systems User Interfaces”; Proc. of ICCS 2004, Krakow, Poland, Lecture Notes in Computer Science 3038 (2004), 505-512.

[Sobecki et al. 06] Sobecki, J., Babiak, E., Slanina, M.: “Application of Hybrid Recommendation in Web-Based Cooking Assistant”; Proc. Of KES 2006, Bournemouth, UK, Lecture Notes in Artifficial Intelligence 4253 (2006), 797-804.

[wikipedia 06] Collaborative Filtering. Download date: 23.05.2006 from http://en.wikipedia.org/wiki/Collaborative_filtering.

[Zlotkin and Rosenschein 93] Zlotkin, G., Rosenschein, J.: “A Domain Theory for Task Oriented Negotiation”; International Joint Conference on Artificial Intelligence (1993), 416-422.